The Fresh Breeze Memory Model Status: Linear Algebra and Plans

Jack Dennis

Joshua Slocum

MIT CSAIL

Guang R. Gao

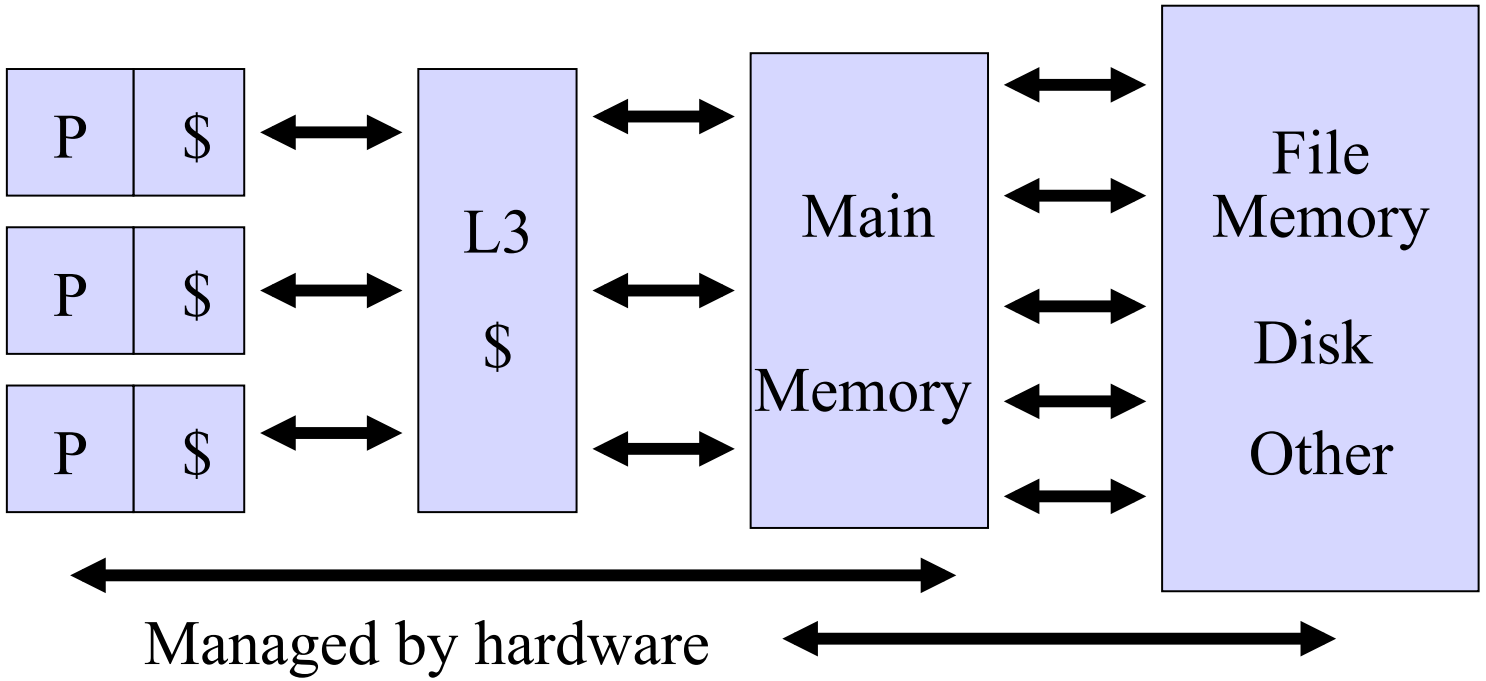
Xiaoxuan Meng

Brian Lucas

University of Delaware

Funded in part by NSF HECURA Grant CCF-0937832

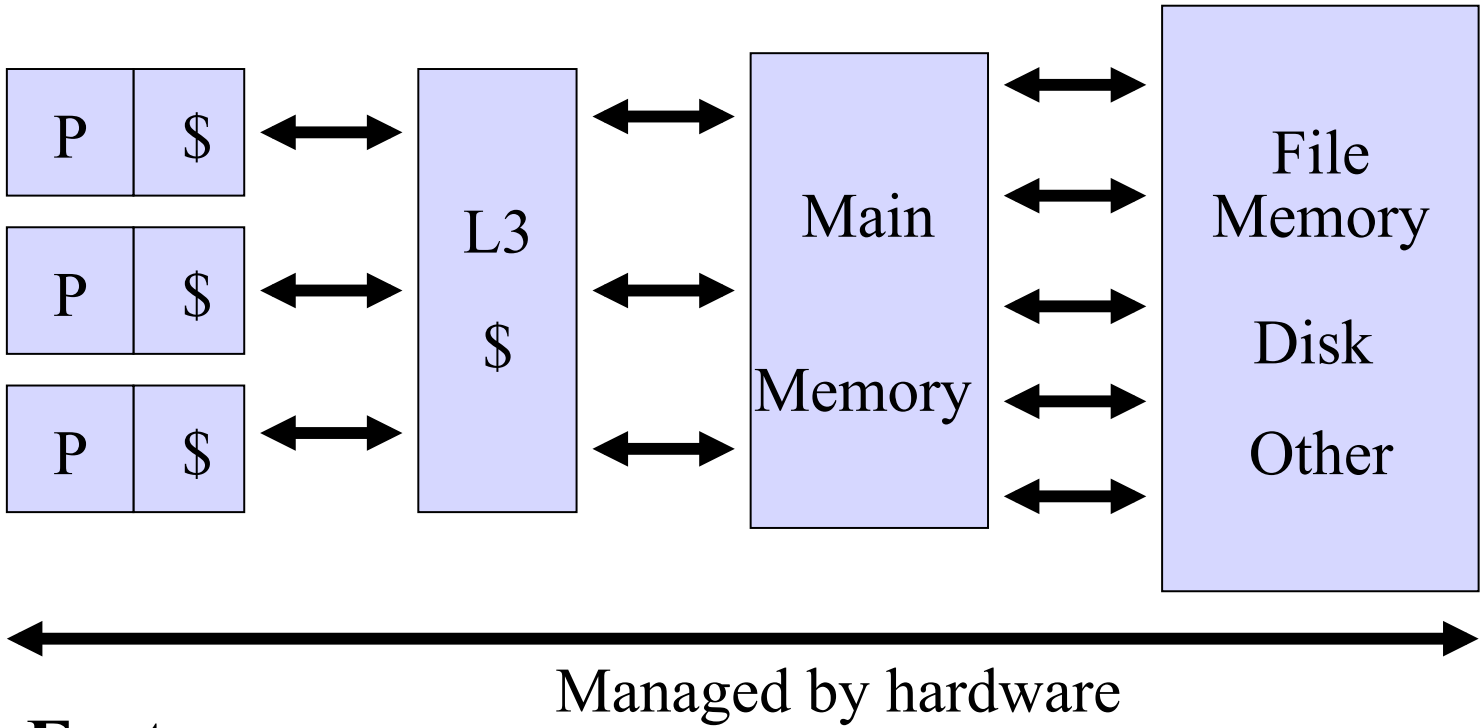
Problem: I/O Performance



Culprits:

- **Operating System Overhead/Noise**
- **Large Units of Data Transfer**
- **Few Concurrent Transfers (OS Limits)**

Solution: Integrated Memory Hierarchy

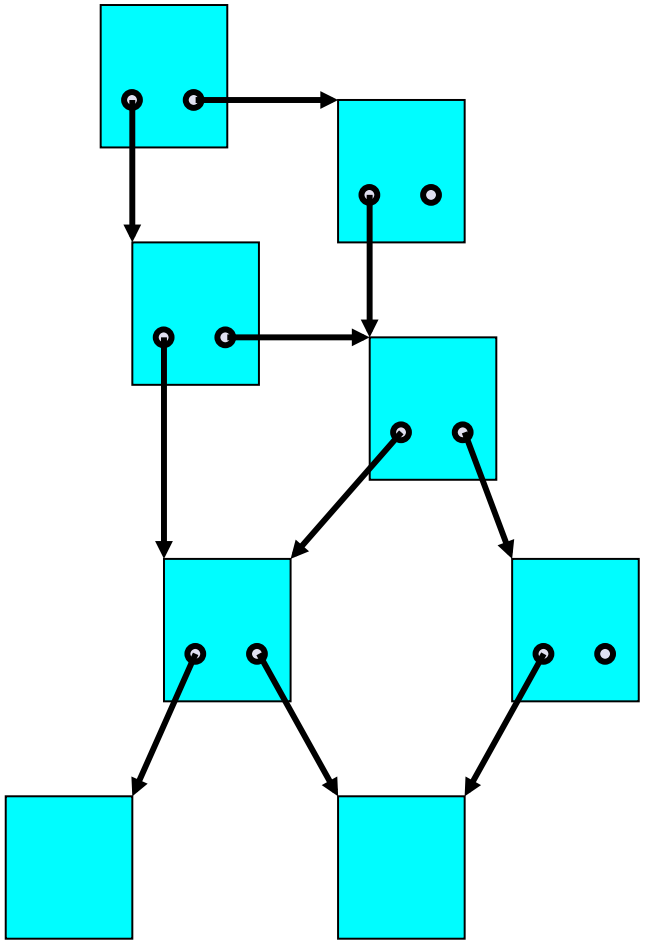


Features:

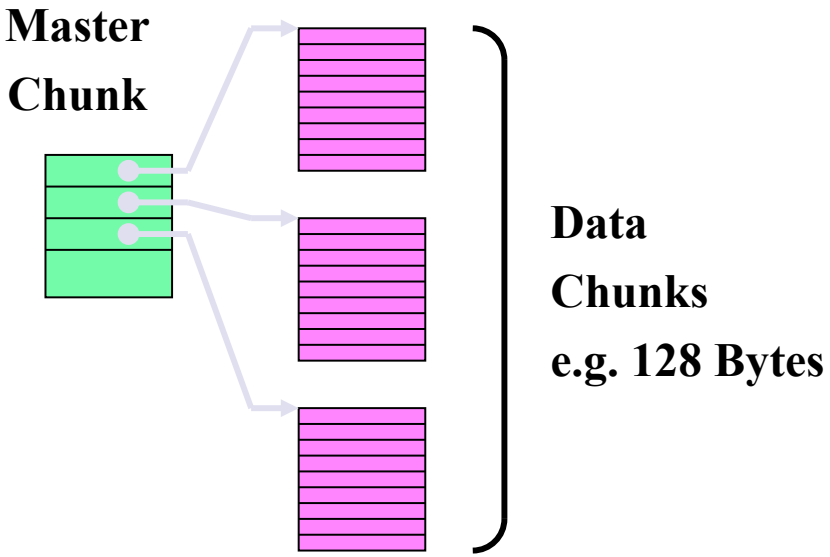
- **Global Virtual Store**
- **Many Concurrent Transactions – High Bandwidth**
- **Superior Basis for Security / Privacy**

The Fresh Breeze Memory Model

Cycle-Free Heap

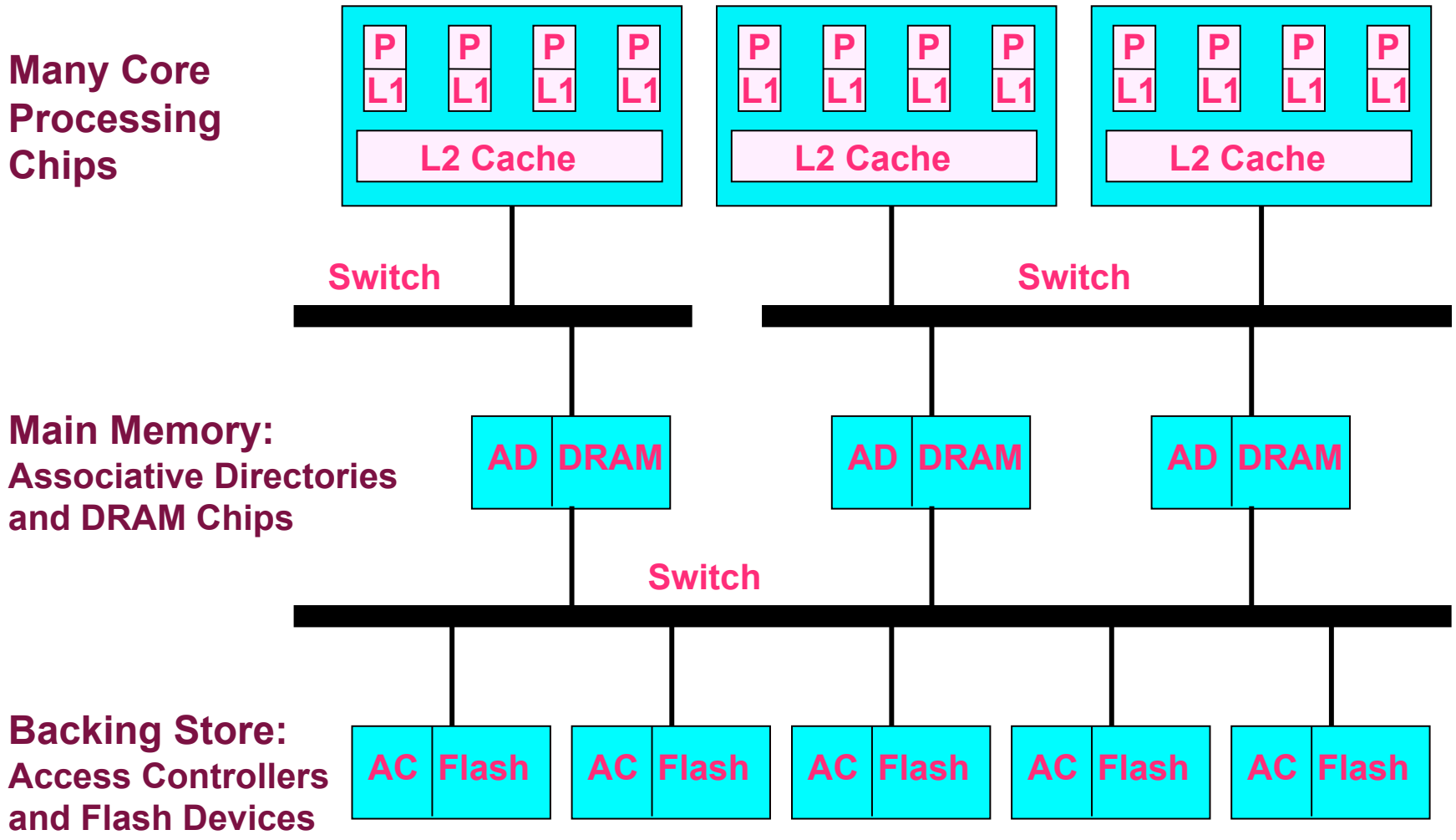


Arrays as Trees of Chunks



- Fan-out as large as 16
- Arrays: Three levels yields 4096 elements (longs)
- Write-Once then Read Only

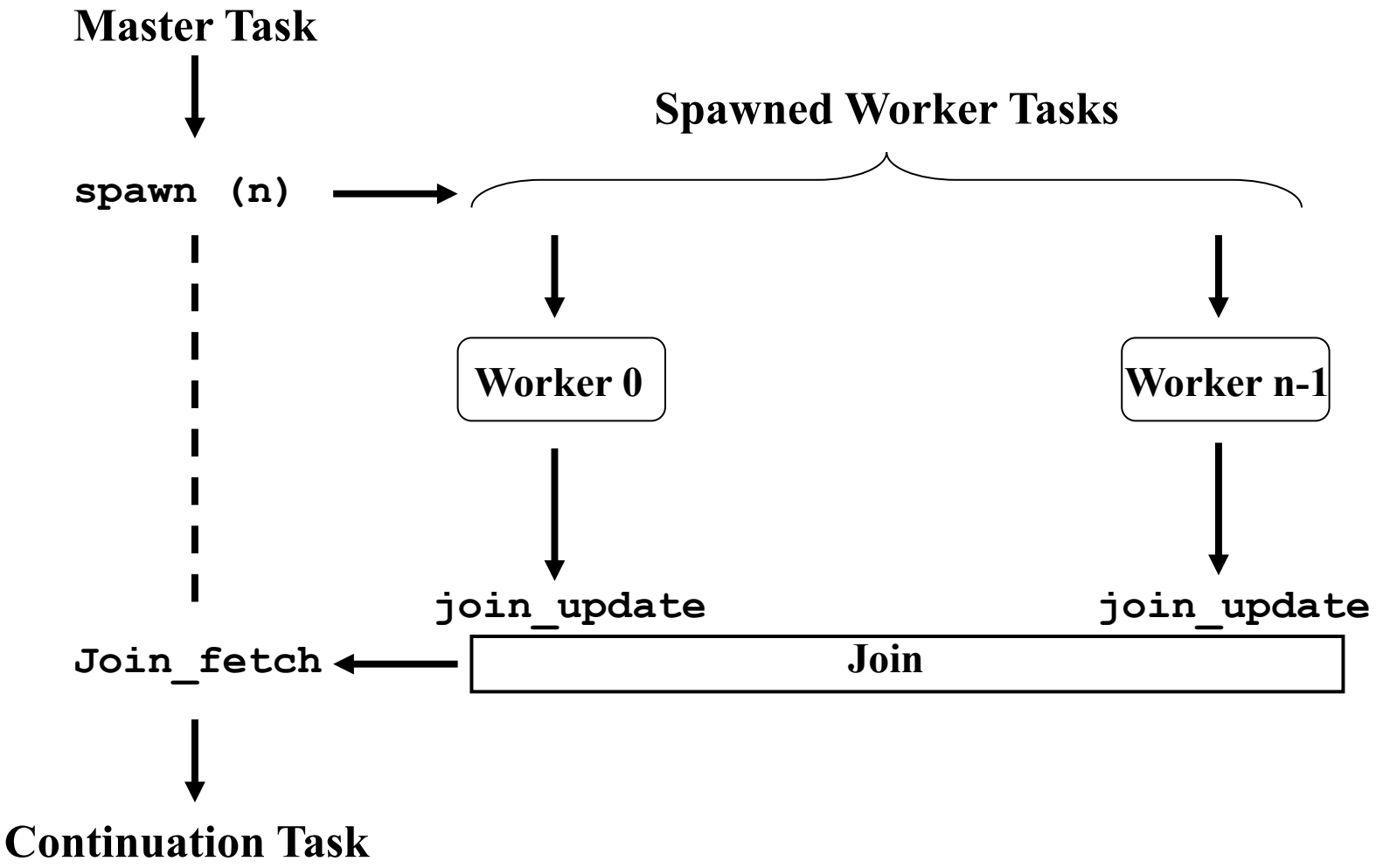
Fresh Breeze System Vision



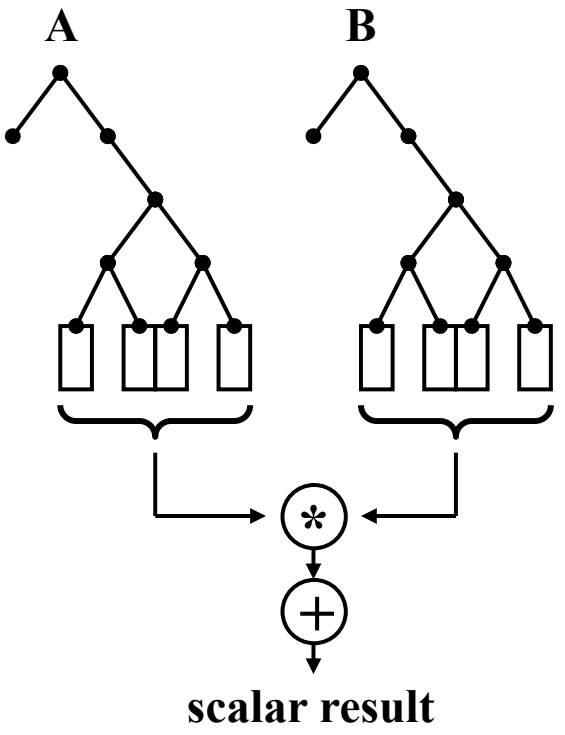
Unique Features of the Processor

- Cache Lines are 128-byte Chunks.
- Registers are tagged to indicate those holding handles of chunks.
- Hardware task scheduler: Active Task List and Pending Task Queue.

Spawn and Join



The Dot Product



5 levels:
Vector length =
16^5 = 1,048,576

Each Leaf Task:
Dot Product of two
16-element vectors:
16 multiplies; 15 adds

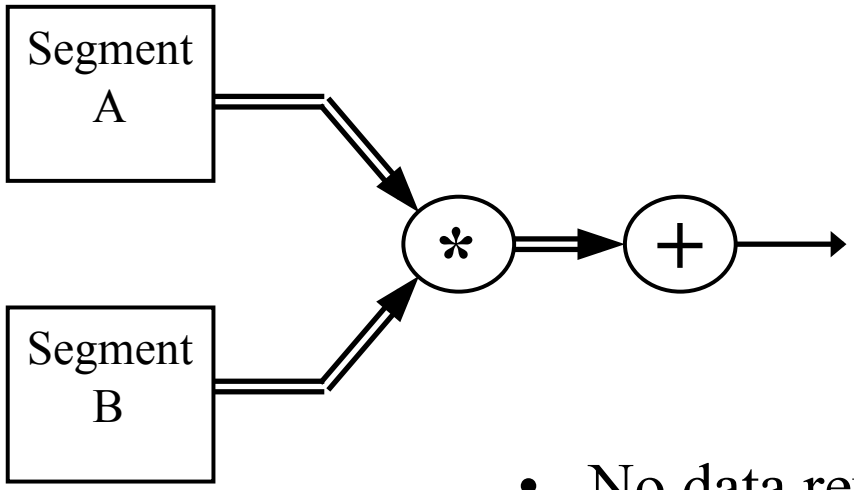
Linear Algebra: Three Algorithms

- Dot Product
- Matrix Multiply
- Fast Fourier Transform

Let's consider the special characteristics of each.

Dot Product

Leaf Task: Dot Product of 16-element segments A and B



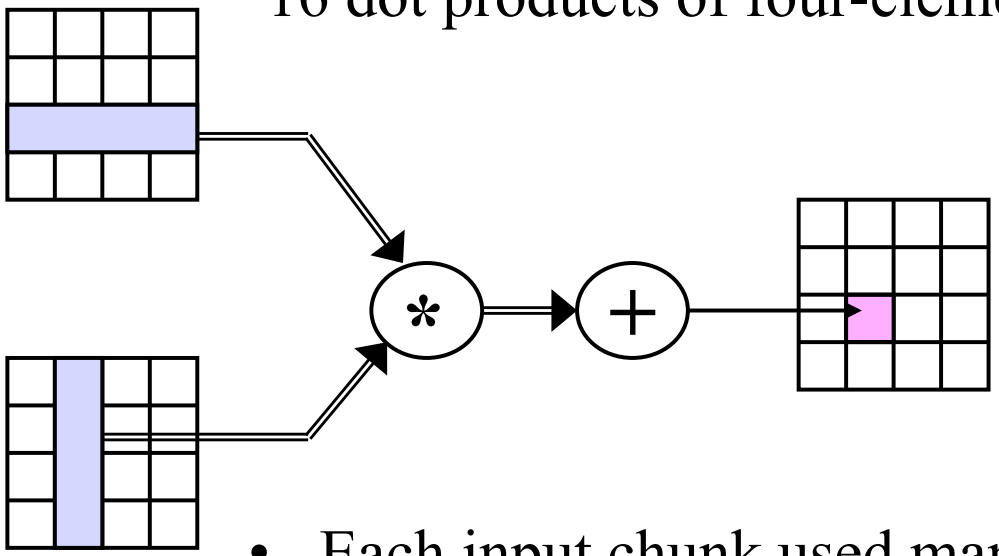
16	Multiplies
15	Adds
31	Operations

- No data reuse
- No intermediate data
- No chunks written
- Large volume of input data

Matrix Multiply

Leaf Task: Product of two 4-by-4 matrices

16 dot products of four-element vectors



64 Multiplies

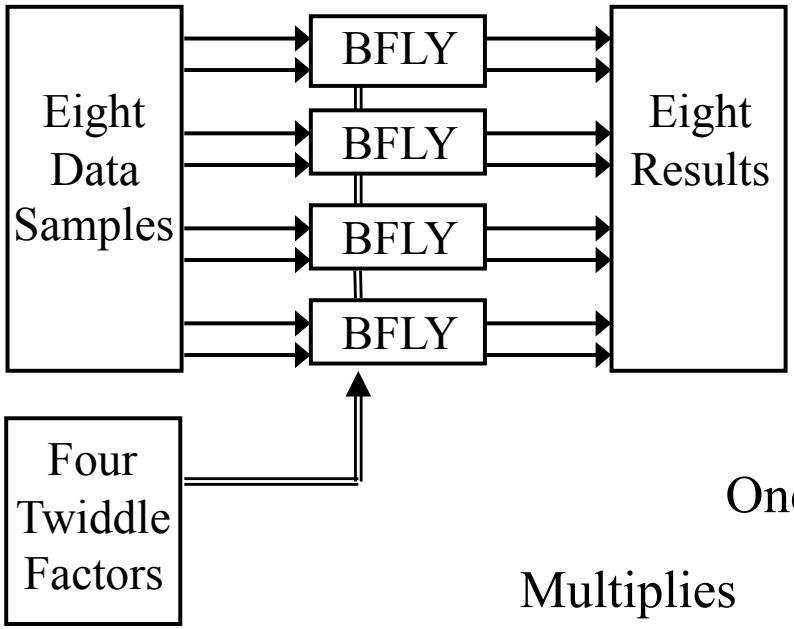
48 Adds

112 Operations

- Each input chunk used many times
- Result chunks written to memory
- No chunks written
- Relatively small input data

Fast Fourier Transform

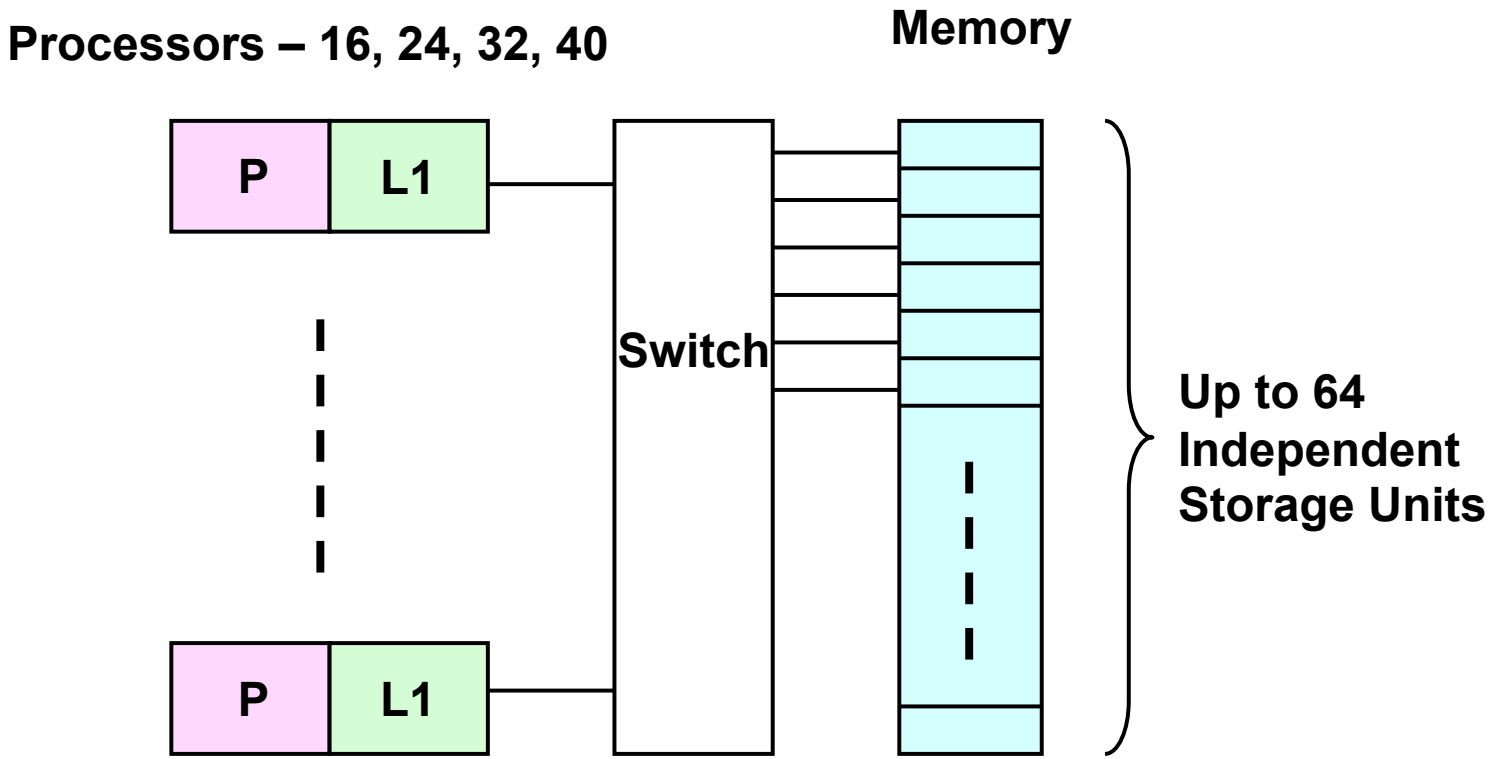
Leaf Task: Group of Four Butterfly Computations



- Log_2 (n) stages
- Intermediate data
- Chunks written and read

	One Butterfly	Four Butterflies
Multiplies	4	16
Adds	6	24
Operations	10	40

Simulated System Model



Simulation Modes

- Non-Blocking:

  A task executing a read simply waits for operation to complete.

  Models an L2 cache with short access time.

- Blocking:

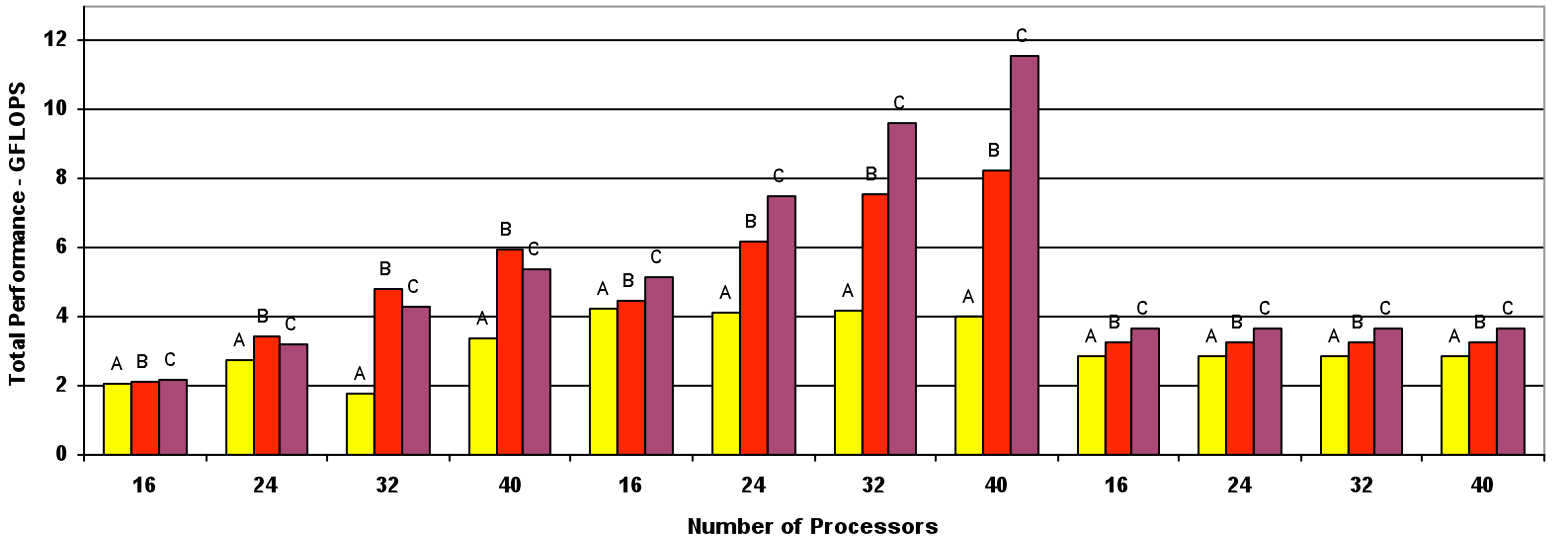
  A task executing a read suspends to permit other tasks to run.

  Models a main memory with long access time.

Estimated Performance

L2 Cache Model - NonBlocking

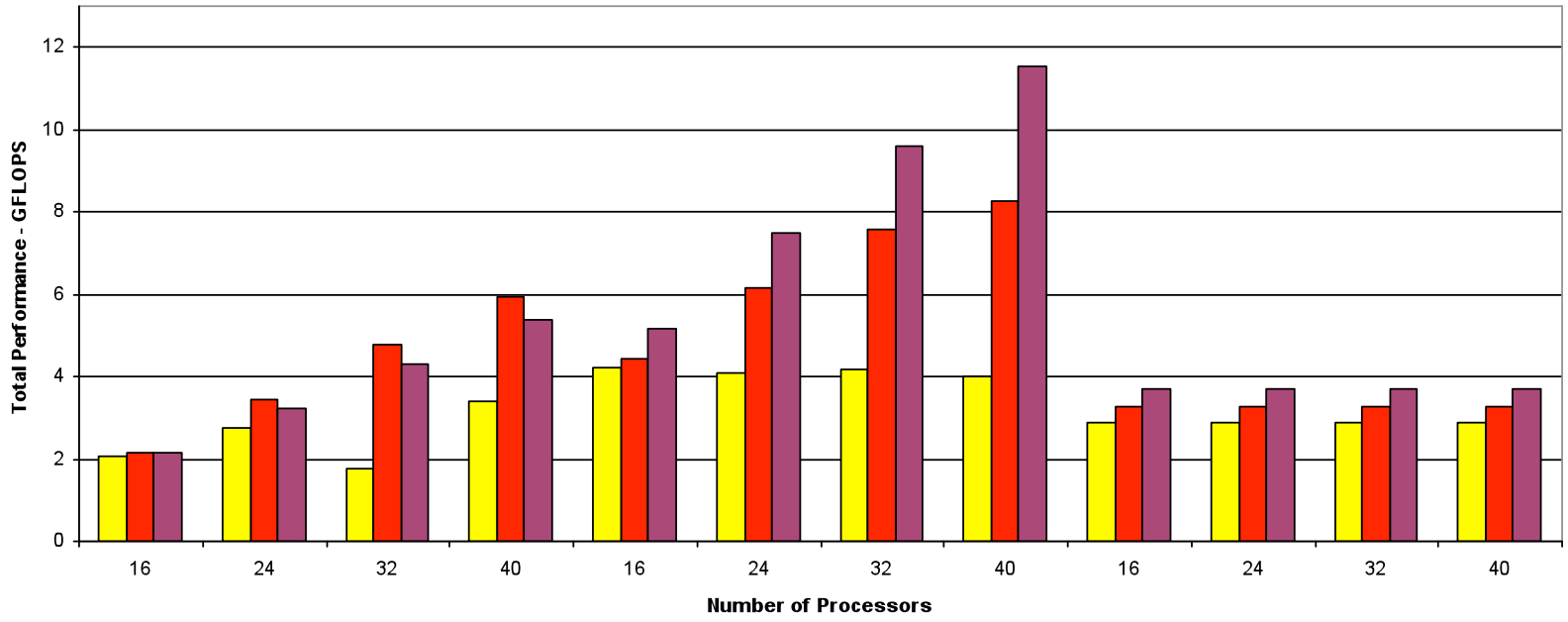
Dot Product	Matrix Multiply	Fourier Transform
A: 16^3	A: 16 x 16	A: 1024
B: 16^4	B: 32 x 32	B: 2048
C: 16^5	C: 48 x 48	C: 4096



Estimated Performance

L2 Cache Model - NonBlocking

Dot Product	Matrix Multiply	Fourier Transfor
A: 16^3	A: 16 x 16	A: 1024
B: 16^4	B: 32 x 32	B: 2048
C: 16^5	C: 48 x 48	C: 4096



Findings

- Data locality of chunks works well.
- L1 Cache is not very important; only a buffer for input and result chunks of tasks.
- Task switching for chunk reads is costly; percolation would make a big difference.

Percolation: Ensuring that input chunks of a task have been retrieved before the task is scheduled.

Further Work

- Model a Three-Level or Four-Level Storage System
- Develop Hierarchical Work Stealing to improve Load Distribution for Massively Parallel Systems.
- Compiler Development: Automatic Mapping of Objects to Trees of Chunks.
- Expand Test Programs to include Transaction Processing and Database Operations .

Distributed Discrete-Event Simulation

- Accurate timing for interconnected communicating components.
- Packet Communication Architecture is a model for distributed systems especially amenable to efficient distributed simulation.
- UDel and MIT have begun a project to develop a PCA-based Simulation Sandbox for evaluating alternate PXMs for massively parallel computing.

Relevance to FSIO

- The Fresh Breeze Memory Model extends to 2^{64} chunks, about 10^{21} chunks or 100,000 exabytes.
- The tree-structure is an excellent basis for advanced security and data object sharing.
- Can simplify check-point / restart.
- Seamless transition from “in-core” to “out-of-core” operation of arbitrary parallel programs.
- Provides ability to use any program as a component of new programs – including any parallel program.

Conclusion

- Making the best of many-core technology requires study of new program execution models (PXMs).
- The FSIO challenge can be met by integrating the file system into the system memory hierarchy as a global virtual store.
- The Fresh Breeze PXM demonstrates some of the benefits from departing from conventional system organization.
- More exploration of new PXMs is needed!